Московский государственный технический университет им. Н.Э. Баумана
Кафедра «Системы обработки информации и управления»

Лабораторная работа №4
по дисциплине
«Методы машинного обучения»
на тему

«Разведочный анализ данных. Исследование и визуализация данных»

Выполнил:
студент группы ИУ5И-21М
Ван Чжэн

Москва — 2024 г.

1. Цель лабораторной работы

изучение продвинутых способов предварительной обработки данных для дальнейшего формирования моделей.

2. Задание

- Реализуйте любой алгоритм семейства Actor-Critic для произвольной среды.

3. Ход выполнения работы

Среда CartPole-v1 - это классическая среда управления, предоставляемая OpenAI Gym, которая также известна как задача инверсии маятника. Эта среда проста в дизайне, но вызывает вызовы и предназначена для тестирования производительности алгоритмов обучения с подкреплением в непрерывных пространствах состояний и действий.

В среде CartPole-v1 есть маленький вагончик (cart), который может двигаться влево и вправо по горизонтальному пути, сверху находится шест (pole). Задача шеста - поддерживать вертикальное положение, не падая. Игрок может управлять вагончиком, прикладывая силу влево или вправо, чтобы попытаться сохранить баланс шеста.

Пространство состояний среды является непрерывным и состоит из четырех значений:

1. Положение вагончика (cart position)
2. Скорость вагончика (cart velocity)
3. Угол шеста (pole angle)
4. Угловая скорость шеста (pole angular velocity)

Игрок может совершать два дискретных действия:

- 0: Приложить силу влево
- 1: Приложить силу вправо

Награда в среде организована следующим образом:

- На каждом временном шаге, если шест все еще стоит, награда равна 1
- Если шест падает или вагончик выходит за границы (превышает ограничения), награда равна 0, и игра заканчивается

Цель среды CartPole-v1 заключается в том, чтобы с помощью подходящей стратегии управления обеспечить как можно более длительное время вертикальное положение шеста, то есть максимизировать накопленную награду.

```
action_dim = env.action_space.n
agent = ActorCritic(state_dim, hidden_dim, action_dim, actor_lr, critic_lr, gamma, device)

return_list = rl_utils.train_on_policy_agent(env, agent, num_episodes)
```

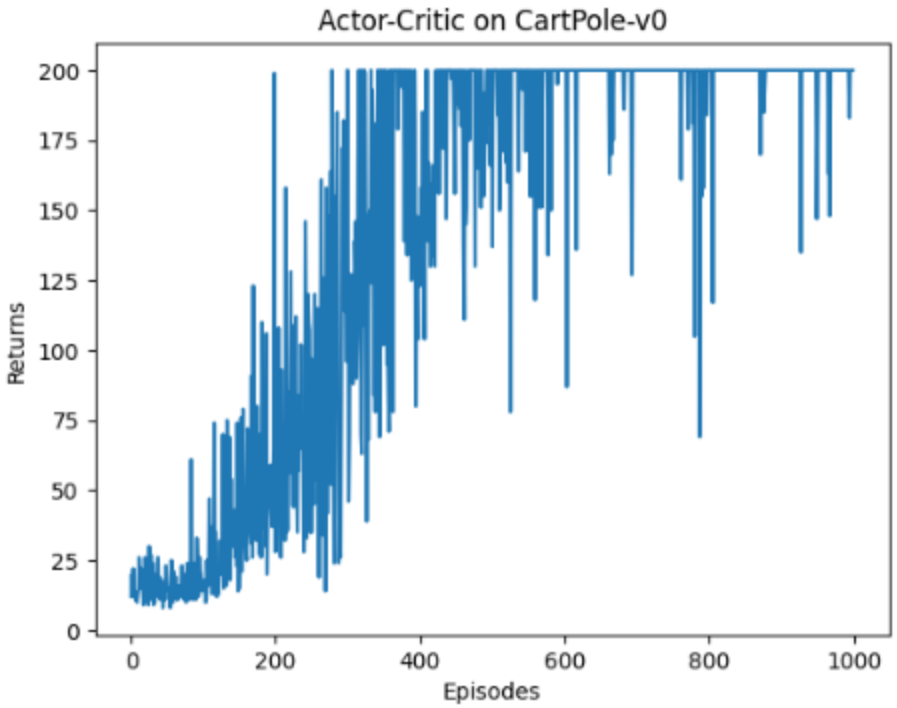
```
/usr/local/lib/python3.10/dist-packages/gym/envs/registration.py:593: UserWarning: WARN: The environment CartPole-v0 is out of date. You should
  logger.warn(
/usr/local/lib/python3.10/dist-packages/gym/core.py:317: DeprecationWarning: WARN: Initializing wrapper in old step API which returns one bool i
  deprecation(
/usr/local/lib/python3.10/dist-packages/gym/wrappers/step_api_compatibility.py:39: DeprecationWarning: WARN: Initializing environment in old ste
  deprecation(
/usr/local/lib/python3.10/dist-packages/gym/core.py:256: DeprecationWarning: WARN: Function `env.seed(seed)` is marked as deprecated and will be
  deprecation(
Iteration 0:   0%|          | 0/100 [00:00<?, ?it/s]<ipython-input-7-00b2d94e3171>:10: UserWarning: Creating a tensor from a list of numpy.ndarr
  state = torch.tensor([state], dtype=torch.float)
/usr/local/lib/python3.10/dist-packages/gym/utils/passive_env_checker.py:241: DeprecationWarning: `np.bool8` is a deprecated alias for `np.bool_
  if not isinstance(terminated, (bool, np.bool8)):
Iteration 0: 100%|██████████| 100/100 [00:01<00:00, 63.03it/s, episode=100, return=19.300]
Iteration 1: 100%|██████████| 100/100 [00:02<00:00, 38.63it/s, episode=200, return=68.500]
Iteration 2: 100%|██████████| 100/100 [00:03<00:00, 26.30it/s, episode=300, return=126.000]
Iteration 3: 100%|██████████| 100/100 [00:07<00:00, 13.91it/s, episode=400, return=147.600]
Iteration 4: 100%|██████████| 100/100 [00:08<00:00, 12.09it/s, episode=500, return=188.900]
Iteration 5: 100%|██████████| 100/100 [00:08<00:00, 11.12it/s, episode=600, return=199.500]
Iteration 6: 100%|██████████| 100/100 [00:09<00:00, 10.30it/s, episode=700, return=192.700]
Iteration 7: 100%|██████████| 100/100 [00:08<00:00, 11.86it/s, episode=800, return=189.700]
Iteration 8: 100%|██████████| 100/100 [00:09<00:00, 10.62it/s, episode=900, return=200.000]
Iteration 9: 100%|██████████| 100/100 [00:09<00:00, 10.80it/s, episode=1000, return=198.300]
```

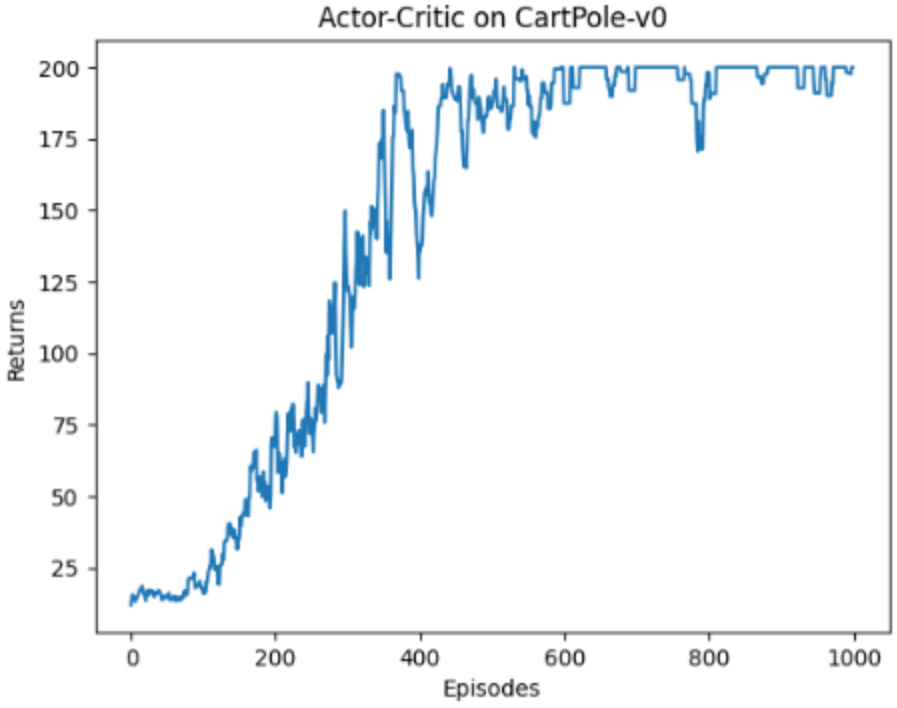
```
episodes_list = list(range(len(return_list)))
plt.plot(episodes_list,return_list)
plt.xlabel('Episodes')
plt.ylabel('Returns')
plt.title('Actor-Critic on {}'.format(env_name))
plt.show()

mv_return = rl_utils.moving_average(return_list, 9)
plt.plot(episodes_list, mv_return)
plt.xlabel('Episodes')
plt.ylabel('Returns')
plt.title('Actor-Critic on {}'.format(env_name))
plt.show()
```


```
episodes_list = list(range(len(return_list)))
plt.plot(episodes_list,return_list)
plt.xlabel('Episodes')
plt.ylabel('Returns')
plt.title('Actor-Critic on {}'.format(env_name))
plt.show()

mv_return = rl_utils.moving_average(return_list, 9)
plt.plot(episodes_list, mv_return)
plt.xlabel('Episodes')
plt.ylabel('Returns')
plt.title('Actor-Critic on {}'.format(env_name))
plt.show()
```


Actor-Critic on CartPole-v0
Returns
Episodes
0
25
50
75
100
125
150
175
200
0
200
400
600
800
1000


Actor-Critic on CartPole-v0
Returns
Episodes
25
50
75
100
125
150
175
200
0
200
400
600
800
1000